

Algorithms for Large, Sparse Network Alignment

Mohsen Bayati, David Gleich, Margot Gerritsen, Amin Saberi,
Ying Wang @ Stanford University

and

Jeong Han Kim @ Yonsei University

Our motivation

Library of Congress subject headings

Desserts

URI: <<http://id.loc.gov/authorities/sh85037243#concept>>

Type: Topical Term

Broader Terms:

- [Confectionery](#)

Narrower Terms:

- [Ambient desserts](#)
- [Banana splits](#)
- [Charlottes \(Desserts\)](#)
- [Chocolate desserts](#)
- [Frozen desserts](#)
- [Ice cream cones](#)
- [Mousses](#)
- [Puddings](#)
- [Refrigerated desserts](#)
- [Sundaes](#)
- [Whipped toppings](#)

LC Classification: TX773

Created: 1986-02-11

Last Modified: 1988-01-15 17:36:44

Wikipedia categories

The screenshot shows a Windows Internet Explorer browser window displaying the Wikipedia article for "Singular value decomposition". The browser's address bar shows the URL http://en.wikipedia.org/wiki/Singular_value_decomposition. The Wikipedia logo is visible on the left side of the page. The article title "Singular value decomposition" is circled in red. Below the title, the text "From Wikipedia, the free encyclopedia" is visible. The article content begins with "In linear algebra, the **singular value decomposition** (SVD) is an important factorization of a rectangular real or complex matrix, with several applications in signal processing and statistics. Applications which employ the SVD include computing the pseudoinverse, least squares fitting of data, matrix approximation, and determining the rank, range and null space of a matrix." A "Contents" section is also visible, listing the following sections: 1 Statement of the theorem, 2 Example, 3 Singular values, singular vectors, and their relation to the SVD, 4 Applications of the SVD, 4.1 Pseudoinverse, 4.2 Solving homogeneous linear equations, 4.3 Total least squares minimization, 4.4 Range, null space and rank, 4.5 Matrix approximation, and 4.6 Separable models.

Singular value decomposition

From Wikipedia, the free encyclopedia

In linear algebra, the **singular value decomposition** (SVD) is an important factorization of a rectangular real or complex matrix, with several applications in signal processing and statistics. Applications which employ the SVD include computing the pseudoinverse, least squares fitting of data, matrix approximation, and determining the rank, range and null space of a matrix.

Contents [hide]

- 1 Statement of the theorem
- 2 Example
- 3 Singular values, singular vectors, and their relation to the SVD
- 4 Applications of the SVD
 - 4.1 Pseudoinverse
 - 4.2 Solving homogeneous linear equations
 - 4.3 Total least squares minimization
 - 4.4 Range, null space and rank
 - 4.5 Matrix approximation
 - 4.6 Separable models

Wikipedia categories

Singular value decomposition - Wikipedia, the free encyclopedia - Windows Internet Explorer

W http://en.wikipedia.org/wiki/Singular_value_decomposition

File Edit View Favorites Tools Help

W Singular value decomposition - Wikipedia, the fre...

Texts and demonstrations [edit]

- MIT Lecture series by Gilbert Strang. See Lecture #29 on the SVD (scroll down to the bottom till you see "Singular Value Decomposition"). The first 17 minutes give the overview. Then Prof. Strang works two examples. Then the last 4 minutes (min 36 to min 40) are a summary. You can probably fast forward the examples, but the first and last are an excellent concise visual presentation of the topic.
- Applications of SVD on PC Hansen's web site.
- Introduction to the Singular Value Decomposition by Todd Will of the University of Wisconsin-La Crosse. This site has animations for the visual minded as well as demonstrations of compression using SVD.
- Los Alamos group's book chapter has helpful gene data analysis examples.
- SVD, another explanation of singular value decomposition
- SVD Tutorial, yet another explanation of SVD. Very intuitive.
- Javascript script demonstrating the SVD more extensively, paste your data from a spreadsheet.
- Chapter from "Numerical Recipes in C" gives more information about implementation and applications of SVD. (Acrobat DRM plug-in required)
- Online Matrix Calculator Performs singular value decomposition of matrices.
- A simple tutorial on SVD and applications of Spectral Methods
- Matrix and Tensor Decompositions in Genomic Signal Processing
- SVD on MathWorld, with image compression as an example application.

Categories: Singular value decomposition | Linear algebra | Matrix theory | Functional analysis

This page was last modified on 22 November 2008, at 17:05. All text is available under the terms of the GNU Free Documentation License. (See [Copyrights](#) for details.)

Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a U.S. registered 501(c)(3) tax-deductible nonprofit charity.

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#)

Powered By Mediawiki

Wikipedia categories

Category:Desserts

From Wikipedia, the free encyclopedia

The main article for this [category](#) is [dessert](#).

Desserts are [sweet foods](#) eaten purely for pleasure, typically at the end of a [meal](#).



Wikimedia Commons has media related to: **[Desserts](#)**

Subcategories

This category has the following 16 subcategories, out of 16 total.

- [\[+\] Desserts by country](#) (20)

B

- [\[+\] Brand name desserts](#) (3)

C

- [\[+\] Cakes](#) (0)
- [\[+\] Chocolate desserts](#) (0)
- [\[+\] Confectionery](#) (9)

C cont.

- [\[+\] Cookies](#) (1)
- [\[+\] Custard desserts](#) (0)

D

- [\[+\] Dessert sauces](#) (0)
- [\[+\] Doughnuts](#) (1)

F

- [\[+\] Frozen desserts](#) (2)

I

- [\[+\] Ice cream](#) (5)

P

- [\[+\] Pastry](#) (4)
- [\[+\] Pies](#) (9)
- [\[+\] Puddings](#) (2)

S

- [\[+\] Sweet breads](#) (2)

μ

- [\[+\] Dessert stubs](#) (1)

Wikipedia vs Library of Congress



Created by **many**,
non-experts, in a
distributed way in a
few years



Library of Congress

Developed by **few**,
experts, in a
centralized way in
over a century.

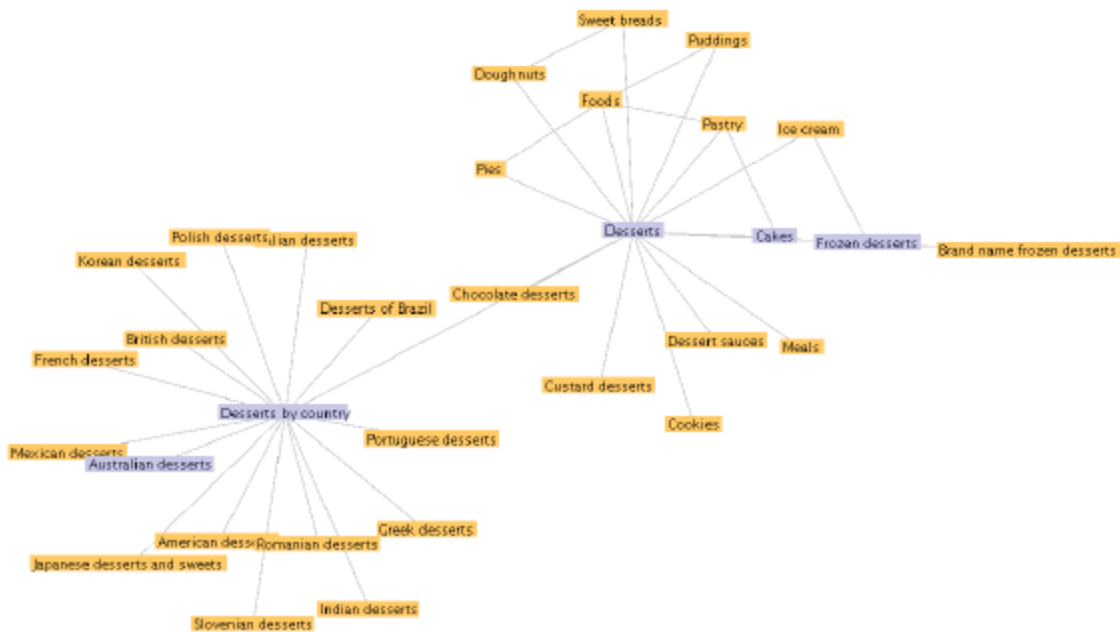
Are they similar ?

Wikipedia vs Library of Congress

- How similar these two data-sets are ?
- Can we use one data-set to enrich the other ?
- How to spend tax-payer's money more wisely to maintain the Library of Congress ?

Project funded by the Library of Congress.

Are these graphs similar



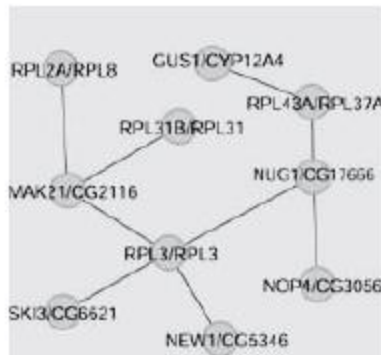
WC



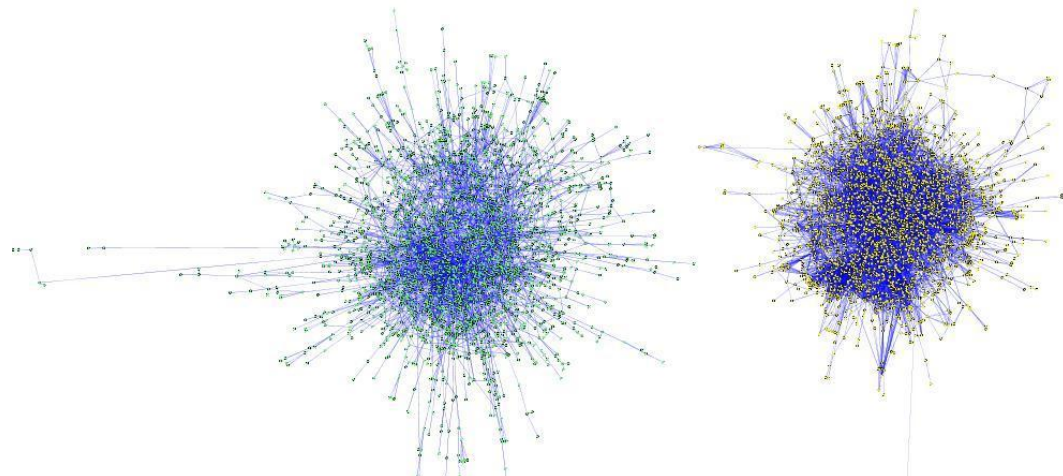
LCSH

Network alignment for Comparing data-sets

- **Match** cross species **vertices** (proteins) and **edges** (protein interactions) → Detect functionally similar proteins.



Berger et al'08, PNAS.

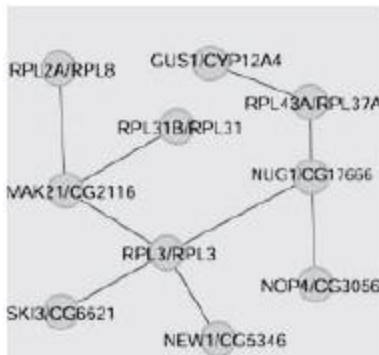


Fly

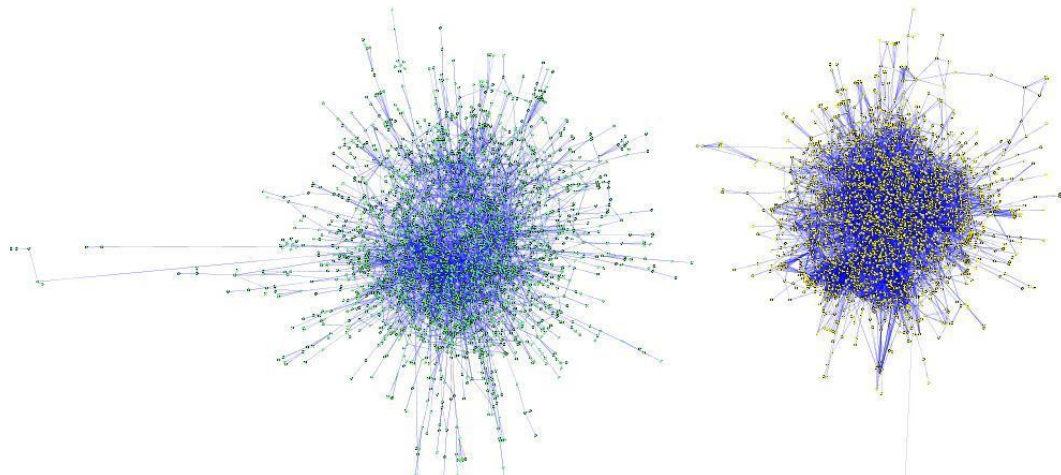
Yeast

Network alignment for Comparing data-sets

- Find the largest common sub-graph on similar vertices. (Singh-Xu-Berger'07,'08).
- Recently (Klau'09).



Berger et al'08, PNAS.



Fly

Yeast

Network alignment for Comparing data-sets

- **Database schema matching**
 - (Melnik-Garcia Molina-Rahm'02).
- **Computer vision:** Match a query image to an existing image.
 - (Conte-Foggia'04)
- **Ontology matching:** Match query image to existing image.
 - (Svab'07).
- **Website :** Match similar parts of the web-graph.
 - Toyota's USA websites vs Toyota France.
- **Social networks:** Teenagers have both fake and real identities.
- **This talk:** Comparing Wikipedia vs Library of Congress.

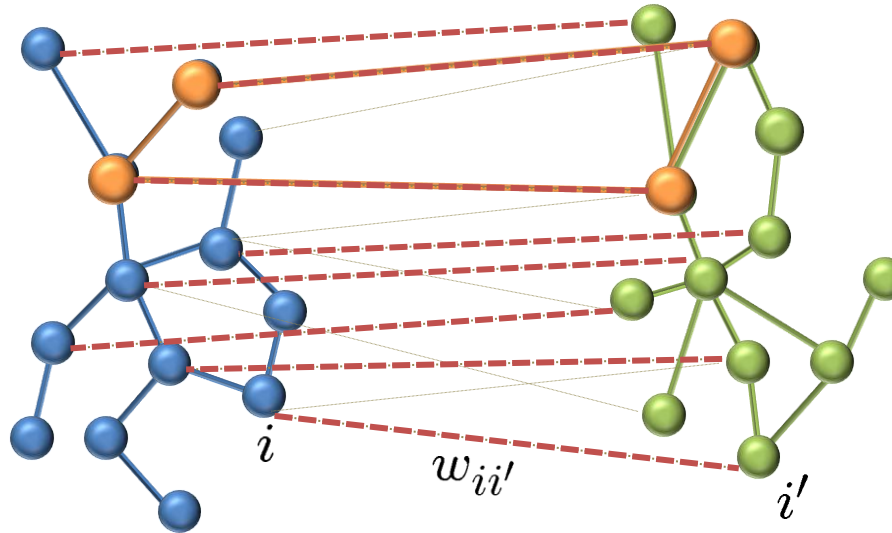
This talk

- Defining the problem mathematically
- Quick survey of existing approaches
- A message-passing algorithm.
- Experiments
 - Real data
 - Synthetic data
- Rigorous results.

Approach: Align the two databases



205,948 nodes
422,503 links



297,266 nodes
248,232 links

5,233,829 potential matches

Goal: Find an **alignment** that **matches similar titles** and **maximizes** the total number of **overlaps**.

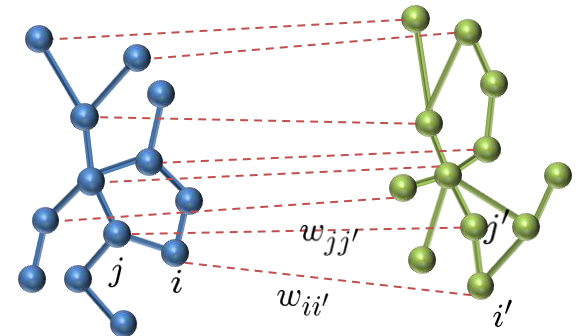
Quadratic program formulation

Formulate the problem as a quadratic program (QP).

$$\begin{aligned} & \text{maximize} && \alpha \sum_{ii'} x_{ii'} w_{ii'} + \beta x^T S x \\ & \text{Subject to:} && \mathbf{A} \vec{x} \preceq \mathbf{1} \\ & && x_{ii'} \in \{0, 1\} \end{aligned}$$

Callouts:

- Total similarity (points to $\alpha \sum_{ii'} x_{ii'} w_{ii'}$)
- Total overlap (points to $\beta x^T S x$)
- Linear constraints (points to $\mathbf{A} \vec{x} \preceq \mathbf{1}$)



Maximizing the similarity alone is easy, but the overlap is NP-hard to maximize. There is a reduction from the MAX-CUT problem.

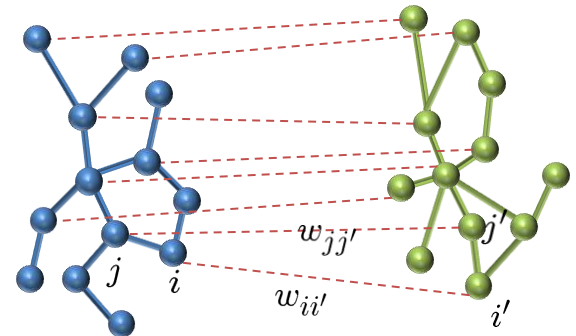
NP-hard to obtain better than 87.8% of the optimum overlap, unless the unique games conjecture is false (Goeman's-Williamson'95).

Quadratic program formulation

maximize $\alpha \sum_{ii'} x_{ii'} w_{ii'} + \beta x^T S x$

Subject to: $\mathbf{A}\vec{x} \preceq \mathbf{1}$

$$x_{ii'} \in \{0, 1\}$$



Related NP-hard problems:

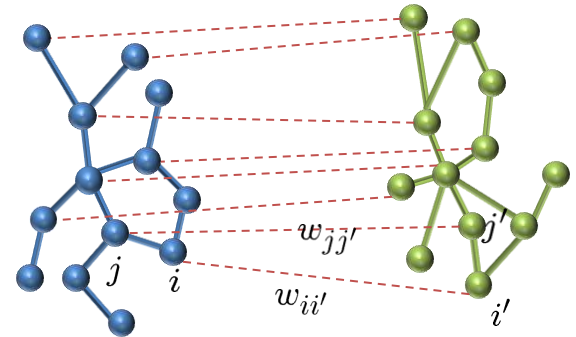
- 1) Maximum common sub-graph.
- 2) Graph isomorphism.
- 3) Maximum clique.

Quadratic program formulation

maximize $\alpha \sum_{ii'} x_{ii'} w_{ii'} + \beta x^T S x$

Subject to: $A\vec{x} \preceq 1$

$$x_{ii'} \in [0, 1]$$



Relaxing the integer constraint \rightarrow Still hard (non-concave max.)

Heuristic 1) Find a local maxima using SNOPT \rightarrow Round to an integer solution.

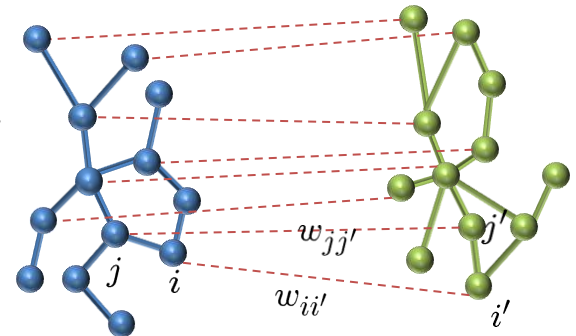
Naïve linear program (LP) formulation

maximize $\alpha \sum_{ii'} x_{ii'} w_{ii'} + \beta \sum_{(ii', jj') \in \mathcal{O}} y_{ii', jj'}$

Subject to: $\mathbf{A}\vec{x} \preceq \mathbf{1}$

$$x_{ii'} \in [0, 1]$$

$$y_{ii', jj'} \leq x_{ii'} \text{ , } y_{ii', jj'} \leq x_{jj'}$$

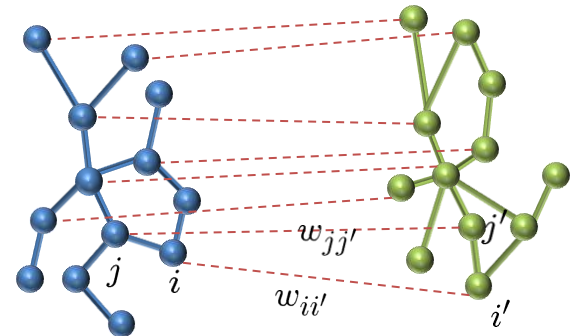


For sparse graphs can be solved relatively efficiently.

Improved LP by Klau'09

$$\begin{aligned} \text{maximize} \quad & \alpha \sum_{ii'} x_{ii'} w_{ii'} + \beta \sum_{(ii'jj') \in \mathcal{O}} y_{ii',jj'} \\ & + \sum_{(ii'jj') \in \mathcal{O}} u_{ii',jj'} \left(y_{ii',jj'} - y_{jj',ii'} \right) \end{aligned}$$

Lagrange multiplier



Subject to:

$$\mathbf{A}\vec{x} \preceq \mathbf{1}$$

$$x_{ii'} \in [0, 1]$$

$$y_{ii',jj'} \leq x_{ii'} \text{ , } y_{ii',jj'} \leq x_{jj'}$$

+ some other combinatorial constraints

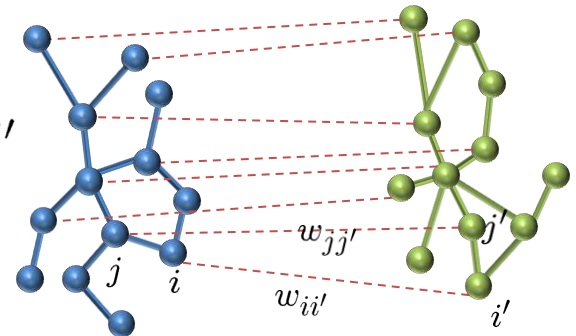
Both LPs and QP also produce an upper-bound for the optimum.

IsoRank (Berger et al'07, 08)

$$\text{maximize } \alpha \sum_{ii'} x_{ii'} r_{ii'} + \beta \sum_{(ii'jj') \in \mathcal{O}} x_{ii'} x_{jj'}$$

$$\text{Subject to: } \mathbf{A}\vec{x} \preceq \mathbf{1}$$

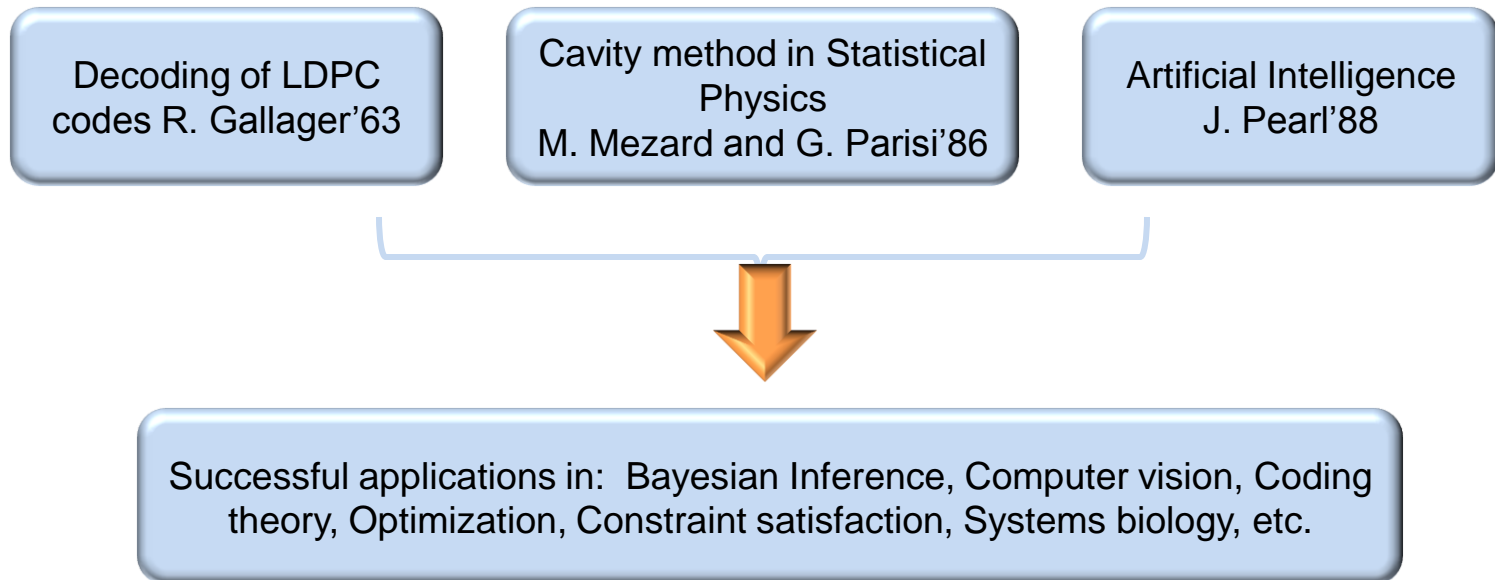
$$x_{ii'} \in [0, 1]$$



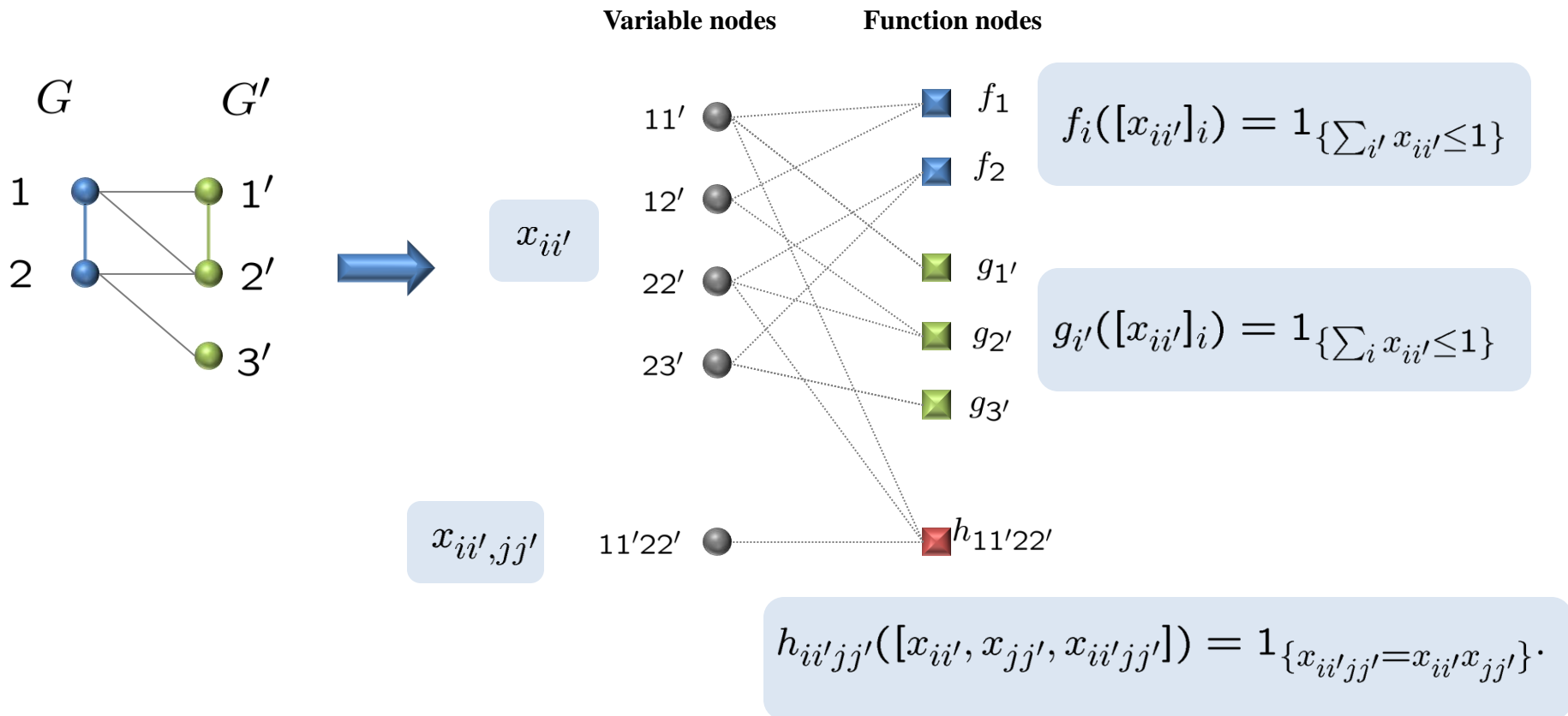
$$r_{ii'} = \sum_{j \in \partial_i} \sum_{j' \in \partial_{i'}} \frac{r_{ii'}}{|\partial_i| |\partial_{i'}|}$$

The new weights, $r_{ii'}$ ts can be found using an eigen-value calculation (similar to PageRank).

Our approach: Belief Propagation (BP)

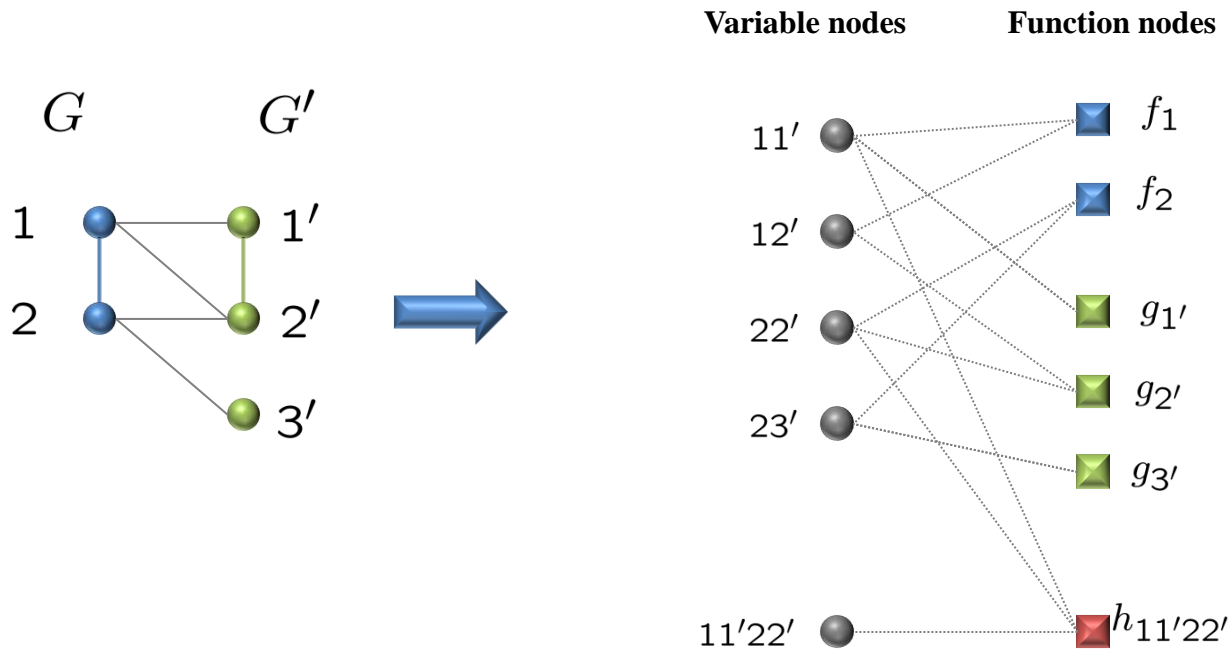


Our approach: Belief Propagation (BP)



Independently, BP was used by Bradde-Braunstein-Mahmoudi-Tira-Weigt-Zecchina'09 for similar problems.

Our approach: Belief Propagation (BP)



$$p(\bar{x}_{E_L}, \bar{x}_S) \propto e^{\alpha \bar{w}^T \bar{x}_{E_L} + \frac{\beta}{2} I^T \bar{x}_S} \prod_{i=1}^n f_i(\bar{x}_{\partial f_i}) \times \prod_{i'=1}^m g_{i'}(\bar{x}_{\partial g_{i'}}) \prod_{ii'jj' \in V_S} h_{ii'jj'}(\bar{x}_{\partial h_{ii'jj'}}).$$

Belief Propagation for $\beta=0$

1) Iterate the following:

For $k = 0, 1, \dots$ update the following messages on each link of the network.

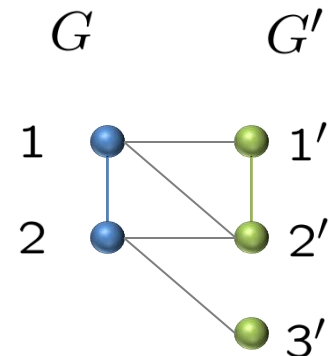
How much i likes to match to i'

$$m_{i \rightarrow i'}^k = w_{ii'} - \max_{j' \neq i'} \left((m_{j' \rightarrow i}^{k-1})^+ \right).$$

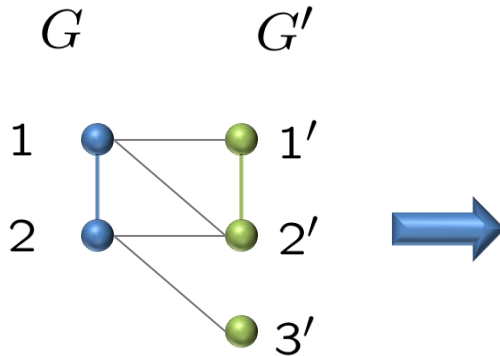
2) The estimated solution at the end of iteration k choose a matching π^k

$$\pi^k(i) = \arg \max_{1 \leq j' \leq n} \left(m_{j' \rightarrow i}^k \right).$$

i.e. pick the link with maximum incoming message.

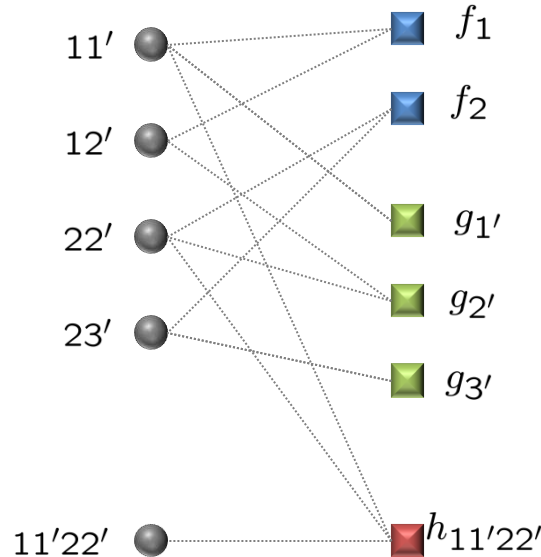


Belief Propagation for $\beta > 0$



Variable nodes

Function nodes



How much i likes to match to i'

$$m_{ii' \rightarrow f_i}^{(t)} = \alpha w_{ii'} - \left(\max_{k \neq i} [m_{ki' \rightarrow g_{i'}}^{(t-1)}] \right)^+ + \sum_{ii'jj' \in O} \min \left(\frac{\beta}{2}, \max(0, \frac{\beta}{2} + m_{jj' \rightarrow h_{ii'jj'}}^{(t-1)}) \right).$$

$$m_{ii' \rightarrow h_{ii'jj'}}^{(t)} = \alpha w_{ii'} - \left(\max_{k \neq i} [m_{ki' \rightarrow g_{i'}}^{(t-1)}] \right)^+ - \left(\max_{k' \neq i'} [m_{ik' \rightarrow f_i}^{(t-1)}] \right)^+ + \sum_{\substack{kk' \neq jj' \\ ii'kk' \in O}} \min \left(\frac{\beta}{2}, \max(0, m_{kk' \rightarrow h_{ii'kk'}}^{(t-1)} + \frac{\beta}{2}) \right)$$

How much ii' likes to have overlap with jj'

Algorithm works for $\beta=0$

(B-Shah-Sharma'05) Each node's decision is correct for $k \geq \lceil \frac{2n \max_{ii'} |w_{ii'}|}{\epsilon} \rceil$

(B-Borgs-Chayes-Zecchina'07): Same algorithm works for any graph when LP relaxation of the problem is integral.

- Generalizes to b-matchings. (independently by Sanghavi-Malioutov-Willskey'07).
- Works for asynchronous updates as well.

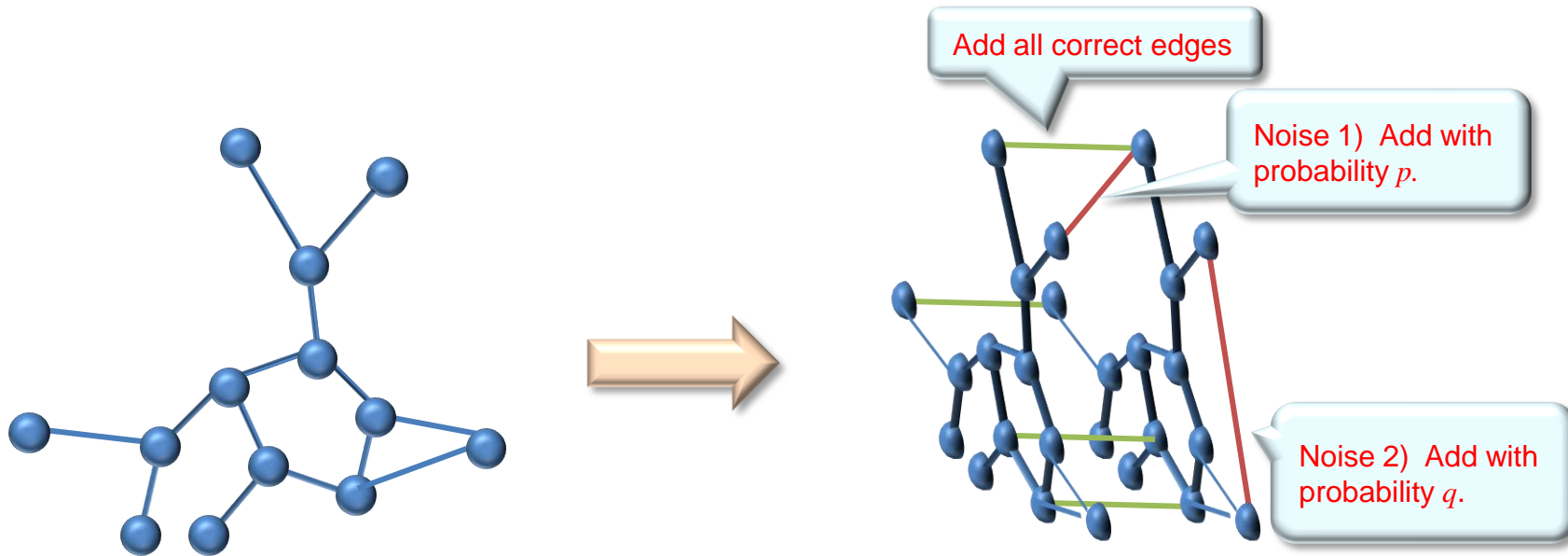
(B-Borgs-Chayes-Zecchina'08): “Belief Propagation” solves the LP relaxation.

- Can use Belief Propagation messages to find the LP solutions in all cases.

How about the $\beta > 0$?

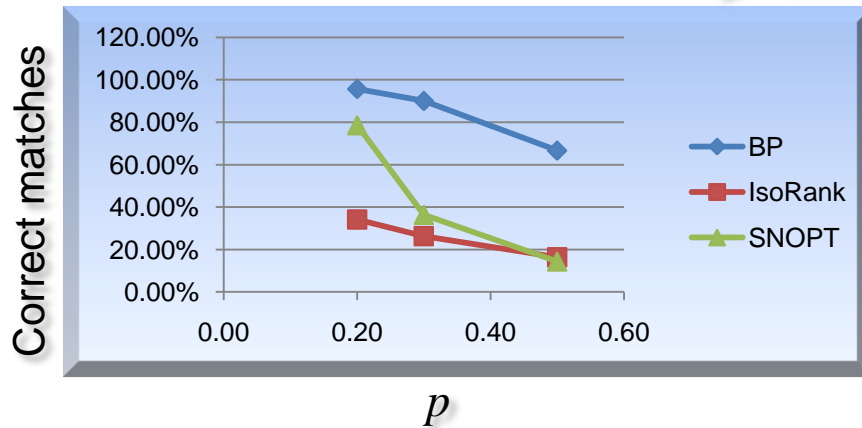
Experiment on Synthetic data

Most of the real-world networks including Wikipedia and LCSH have power-law distribution (The node degree distribution satisfies $P(d_i = k) = \frac{1}{k^\theta}$.)



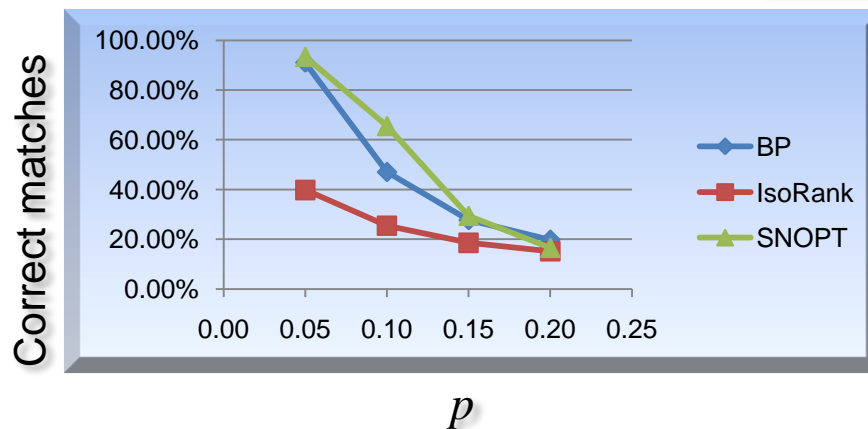
Experiment on Synthetic data

$q = 0$



BP, IsoRank → few seconds
SNOPT → few hours

$q = 0.2$

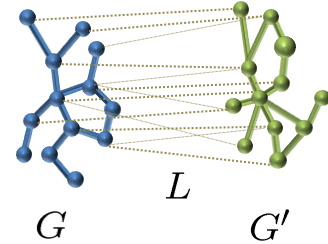
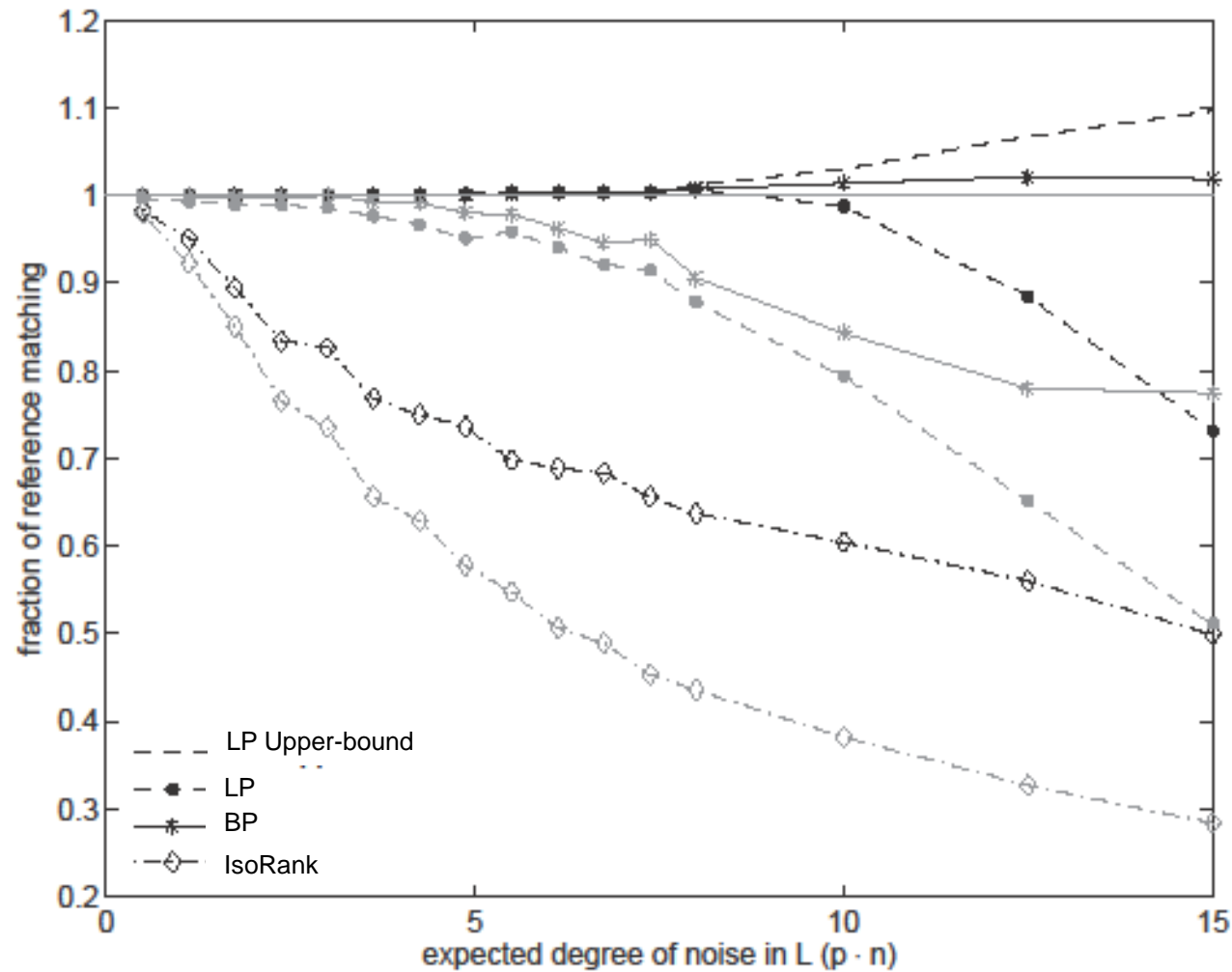


Add all correct edges

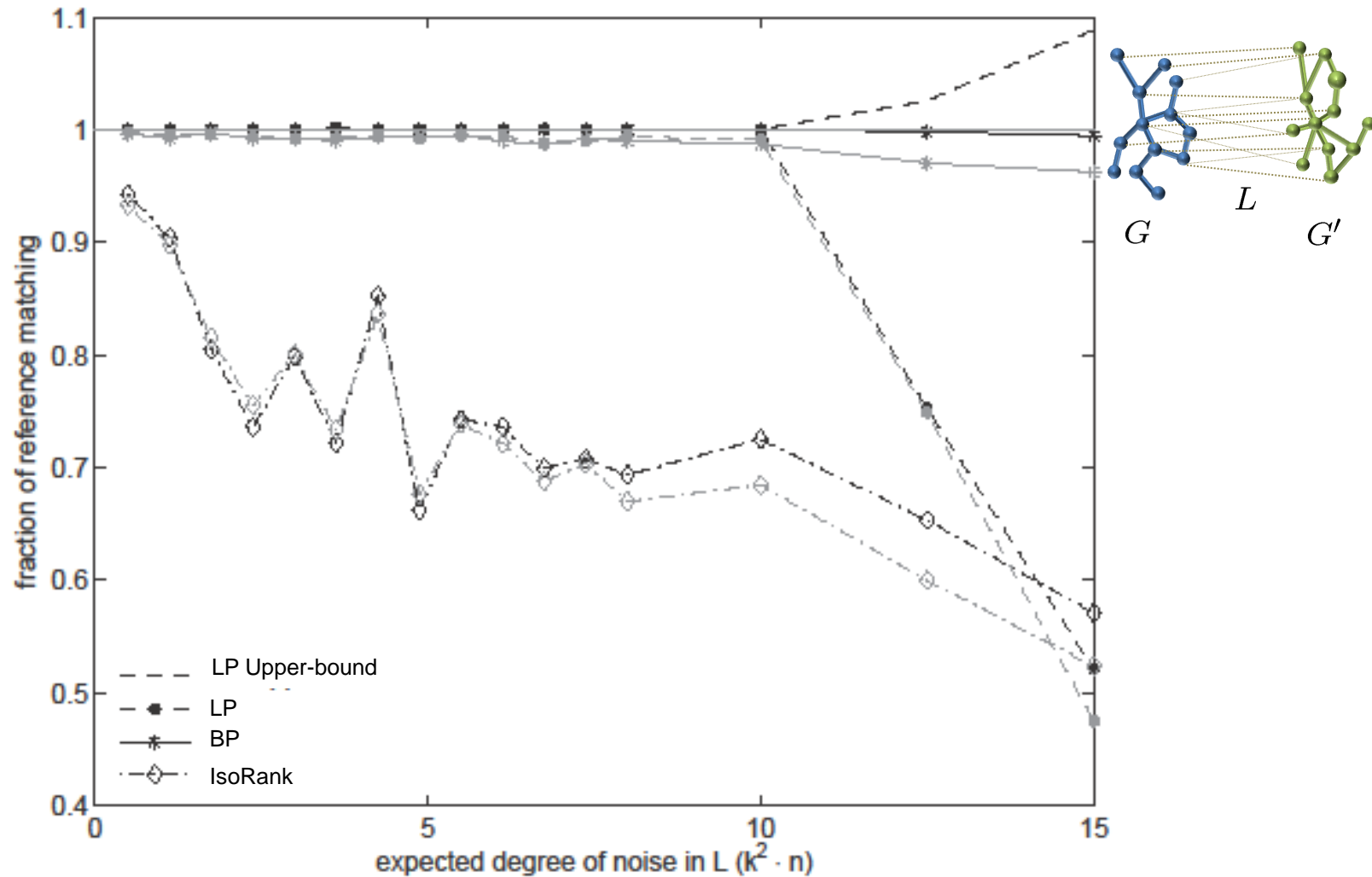
Noise 1) Add with probability p .

Noise 2) Add with probability q .

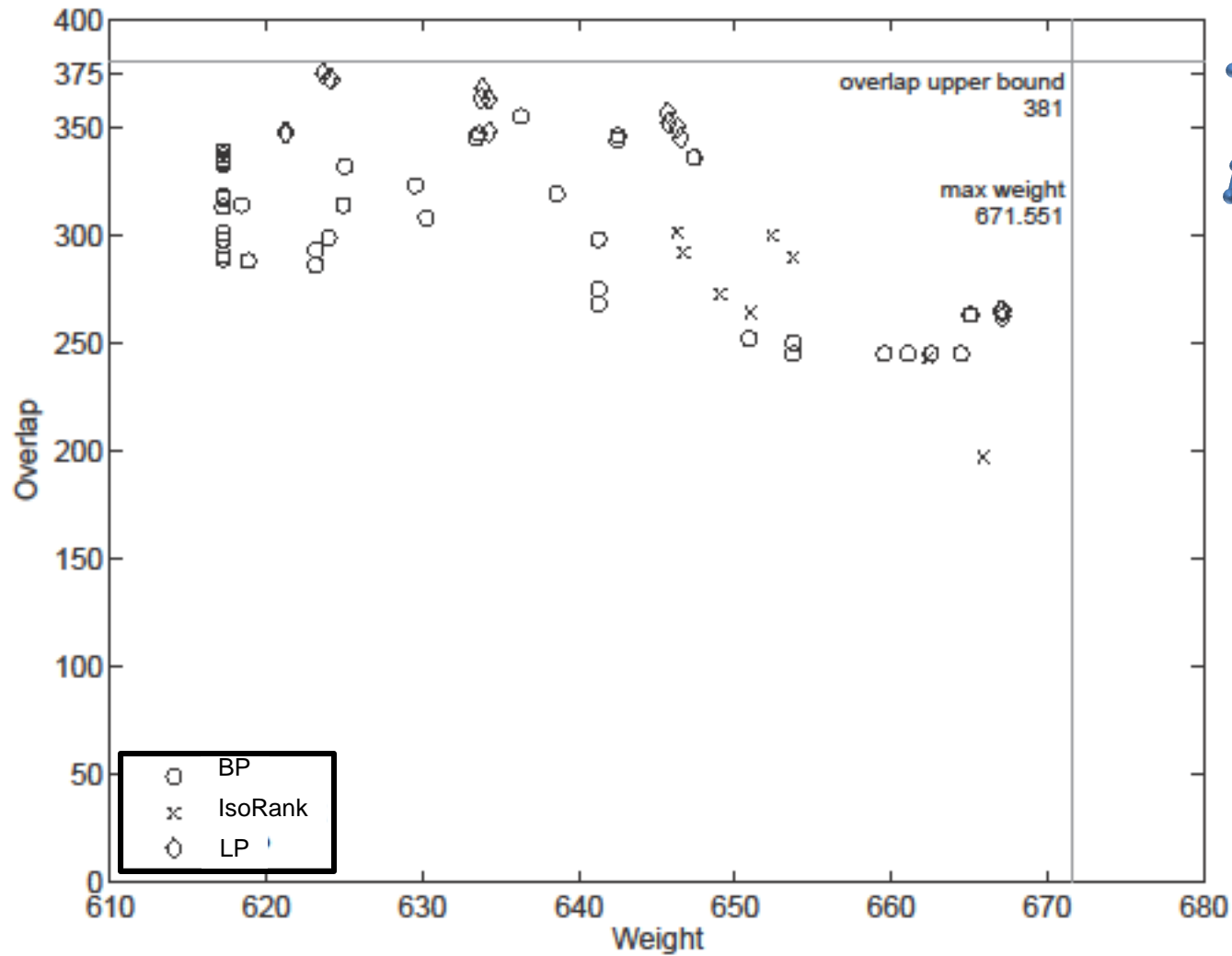
Power-law graph experiments



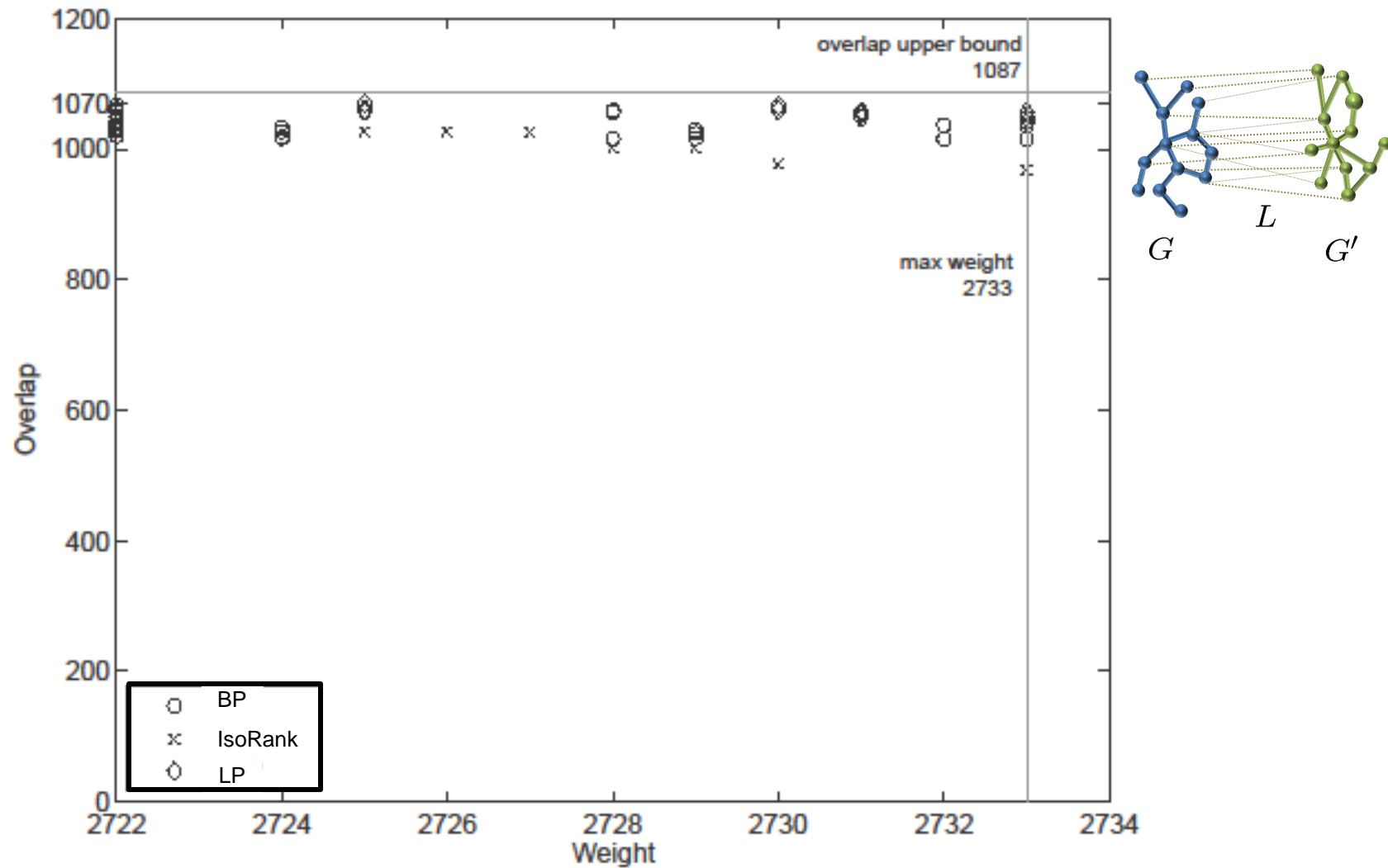
Grid graphs experiments



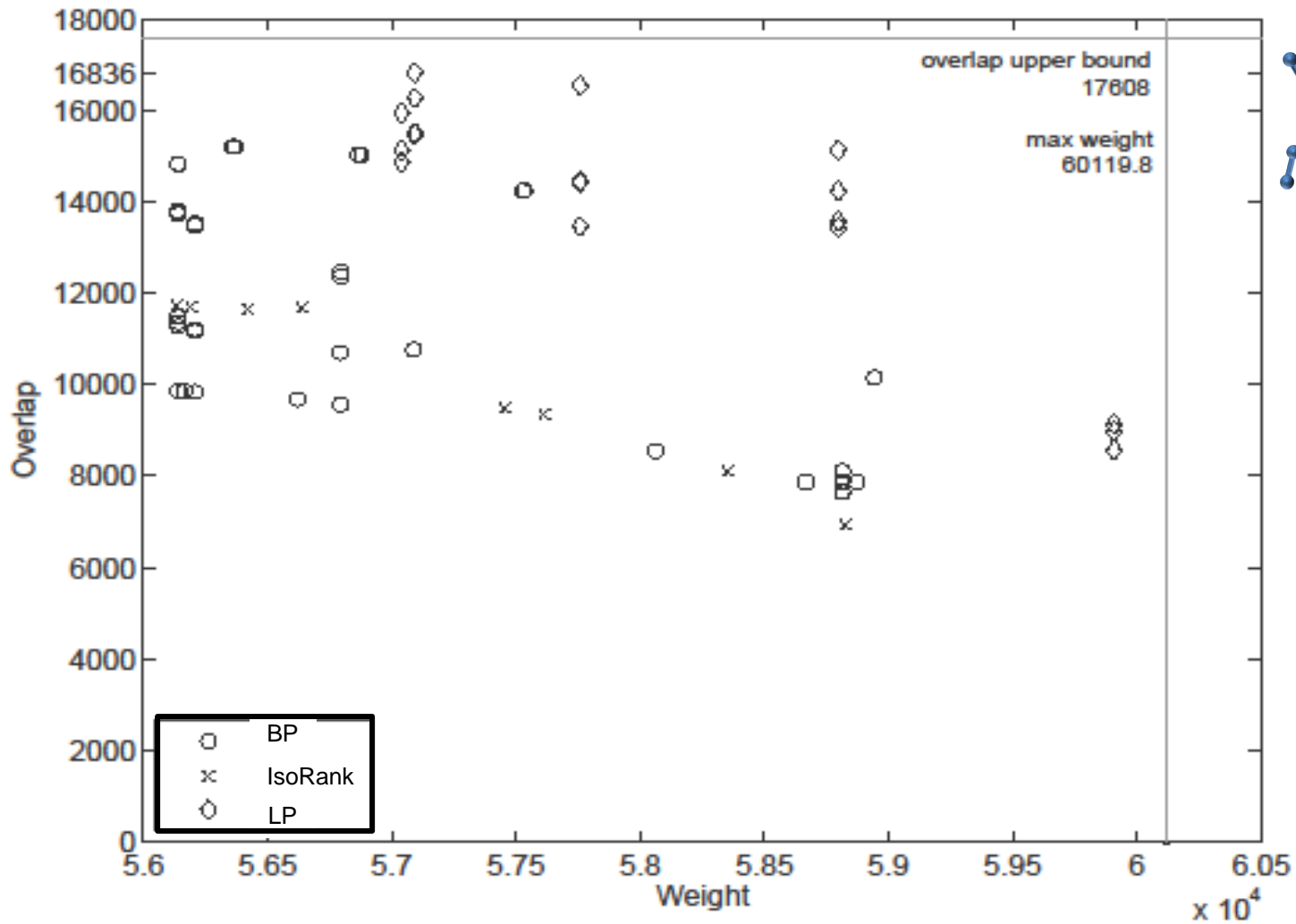
Bioinformatics data: Fly vs Yeast



Bioinformatics data: Human vs Mouse



Ontology data: Wiki vs LCSH

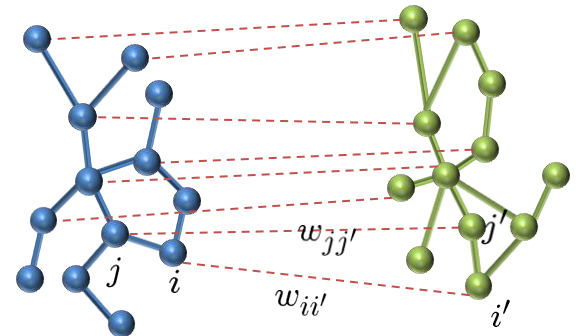


Statistical significance

maximize $\alpha \sum_{ii'} x_{ii'} w_{ii'} + \beta x^T S x$

Subject to: $A\vec{x} \preceq 1$

$$x_{ii'} \in \{0, 1\}$$



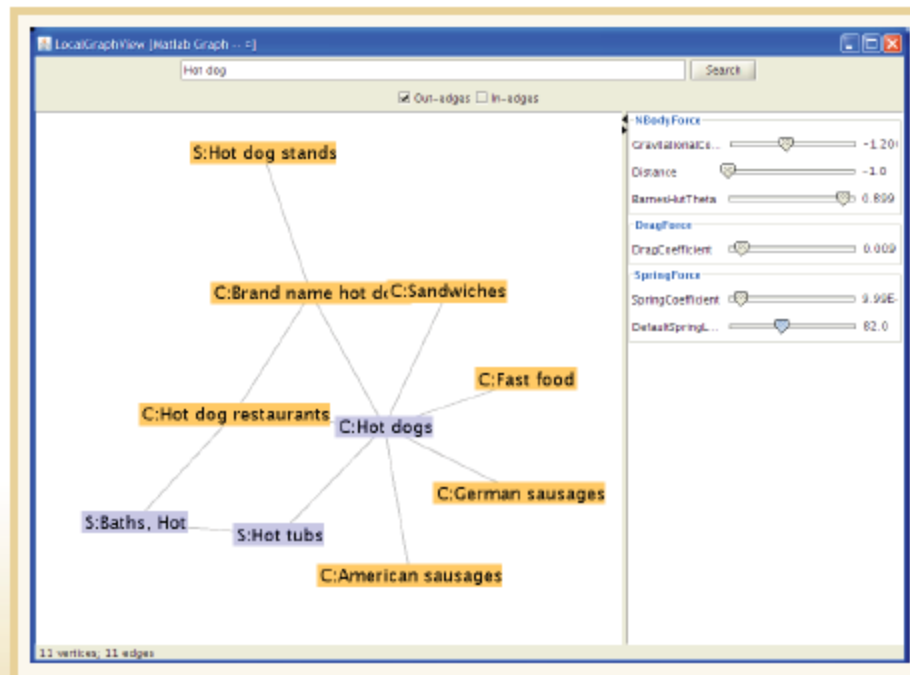
Create many uniform random samples of LCSH and Wiki with the same node degrees. The objective value drops by 99%.



Statistical evidence that
the two data-sets are
very comparable.

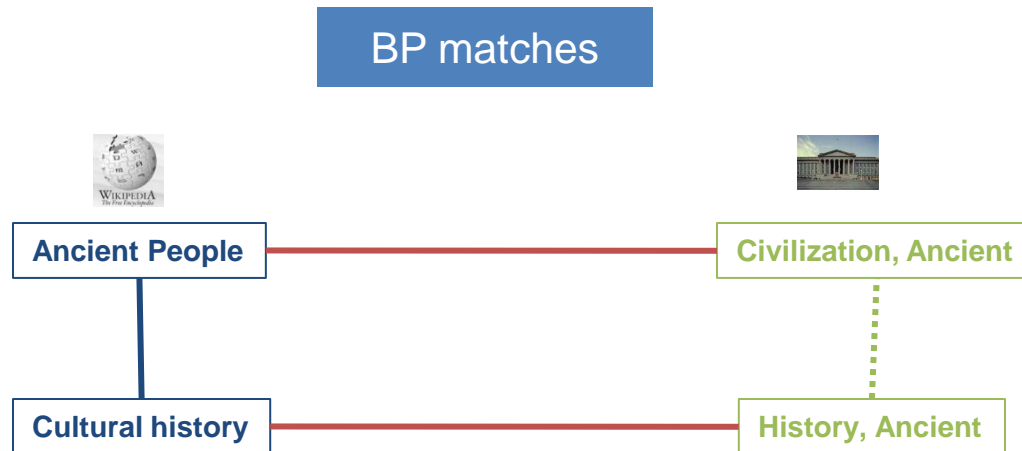
Some matched titles

LCSH	WC
Science fiction television series	Science fiction television programs
Turing test	Turing test
Maching learning	Machine learning
Hot tubs	Hot dog



Enriching the data-sets

- The approach suggests few thousands of potential links to be tested with human experts in the Library of Congress.



Conclusions

- Only BP, IsoRank and LP can handle large graphs.
- BP and LP find near optimum solution on sparse data
- LP produces an upper bound, and slightly better results. But slightly slower.
- For denser graphs BP outperforms LP.

Thank You!